Reachability Criterion with Sufficient Test Space for Ordinary Petri Net

Gi Bum Lee, Han Zandong and Jin S. Lee

1. Introduction

Petri nets (PN) are widely recognized as a powerful tool for modelling and analyzing discrete event systems, especially systems are characterized by synchronization, concurrency, parallelism and resource sharing [1, 2]. One of the major advantages of using Petri net models is that the PN model can be used for the analysis of behaviour properties and performance evaluation, as well as for systematic construction of discrete-event simulators and controllers [3, 4]. The reachability from an initial marking to a destination marking is the most important issue for the analysis of Petri nets. Many other problems such as liveness and coverability can be deduced from this reachability problem [5, 6].

Two basic approaches are usually applied to solve the reachability problem. One is the construction of reachability tree [7, 8]. It can obtain all the reachable markings, but the computation complexity is exponentially increased with the size of a PN. The other is to solve the state equation [9]. The solution of the matrix equation provides a firing count vector that describes the relation between initial marking and reachable markings. Its major problem is the lack of information of firing sequences and the existence of spurious solutions.

Many researchers have investigated the reachability problem [10, 11]. Iko Miyazawa et al. have utilized the state equation to solve the reachability problem of Petri nets with parallel structures [12]. Tadashi Matsumoto et al. have presented a formal necessary and sufficient condition on reachability of general Petri nets with known firing count vectors [13]. Tadao Murata’s paper has concentrated on presenting and analyzing Petri nets as discrete time systems. Controllability and reachability are analyzed in terms of the matrix representation of a Petri net [14].

In most cases, it is not necessary to find all reachable markings. One of the most important things is to know whether a given marking is reachable or not. If the destination marking
is reachable from the initial marking \(M_0\), it is significant to find a firing sequence, which is an ordered sequence of transitions that lead \(M_0\) to \(M_d\). The following method can be utilized to find a reachable marking [15].

i. Solve the equation \(AX = M_d - M_0\) to ascertain all the solutions \(X_1, X_2, \ldots\) and construct the set \(X = \{X_1, X_2, \ldots\}\).

ii. Test if \(X_i\) in \(X\) is an executable solution from \(M_0\), i.e. there is at least one sequence \(S(X_i)\) that is a firing sequence under \(M_0\).

iii. If an executable solution exists, then \(M_d\) is reachable. On the contrary, if \(X = \emptyset\) or all solutions are spurious, then \(M_d\) is not reachable.

However, this approach is theoretic rather than practical, because there are two problems: One is that the solution of the fundamental equation \(AX = M_d - M_0\) is infinite in some cases. In that case, it is impossible to test all solution \(X_i\). The other is that the computation complexity of testing \(X_i\) increases at least exponentially as the length of \(S(X_i)\) increases.

In this chapter, the above two problems will be solved as follows: First, we construct a sufficient test space to include at least one executable solution within set \(X\). An approach is secondly proposed to test whether there is an executable solution within the sufficient test space or not. A systematic method to search an executable solution in a sufficient test space and to enumerate the associated firing sequence is presented.

The remainder of the chapter is arranged as follows: Definitions and notations required in this chapter are given in Section 2. Section 3 describes how to determine the sufficient test space for the reachability problem. In Section 4, an algorithm is developed to determine if \(X_i\) is a executable solution under \(M_0\) and gives the associated firing sequence \(S(X_i)\). The illustrative examples are given in Section 3, Section 4, and Section 5.

### 2. Preliminaries

In this section, we present some definitions and notations to be necessary in the following sections.

**Definition 1.** Let \(PN=(P, T, I, O, M_0)\) be a marked Petri net. \(P=\{p_1, p_2, \ldots, p_n\}\) is the finite set of places. \(T=\{t_1, t_2, \ldots, t_m\}\) is the finite set of transitions. \(I\) is the input function. \(O\) is the output function. \(M_0\) is the initial marking.

A PN is an ordinary Petri net iff \(I(p, t)\to\{0, 1\}\) and \(O(t, p)\to\{0, 1\}\) for any \(p\in P\) and \(t\in T\). \(A = O - I\) is the incidence matrix, where \(O\) and \(I\) are the output and input function matrices [16]. Let \(X = [x_1 x_2 \ldots x_m]^T\) be a column vector. If \(X\) is the firing count vector of \(S(X)\), the sequence \(S(X)\) is called the transition sequence associated with \(X\). The transition set \(T(X)\) is called the support of \(X\) if it is composed of transitions associated with positive elements of \(X\), i.e. \(T(X) = \{t| x_t > 0\}\). \(p^o\) is the set of output transitions of \(p\), \(p^i\) is the set of input transitions of \(p\), \(t^o\) is the set of output places of \(t\), and \(t^i\) is the set of input places of \(t\).

**Definition 2.** \(C = \langle p, T_o \rangle\) is called a conflict structure [17] if it satisfies the following condition: \(T_o = \{t| t \in p^o\}\) and \(|T_o| \geq 2\), where \(|T_o|\) is the cardinality of \(T_o\). We note that \(C = \{C_1, C_2, \ldots\}\) is the set of all \(C_i\) and \(T_o = T_{o1} \cup T_{o2} \cup \ldots\) is the set of all conflict transitions.
Definition 3. For transition $t_j$ and $X$, the sub-vector $H(t_j | X)$ is defined as: $H(t_j | X) = e[t_j] \cdot x_j$, where $e[t_j]$ is the unit m-vector which is zero everywhere except in the $j$-th element.

Definition 4. For the conflict structure $C_i=\langle p, T_{ci} \rangle$ and $X$, the sub-vector $H(C_i | X)$ is defined as follows:

$$H(C_i | X) = \sum_{t_j \in T_{ci}} H(t_j | X)$$

(1)

Definition 5. $C_i=\langle p, T_{ci} \rangle$ is in a spurious conflict state for $X$ under $M$ if there exists a firing sequence $S(H(C_i | X))$ under $M$, i.e. the mathematic criterion is $M \geq I \cdot H(C_i | X)$.

Otherwise, $C_i$ is in an effective conflict state for $X$ under $M$, and the transition in $T_{ci}$ is called the effective conflict transition for $X$ under $M$.

Notation 1. $N(t_j | S(X)) = x_j$ is the number of occurrence times of $t_j$ in $S(X)$.

Notation 2. If $q = \min\{M(p_i), p_i \in \delta(t_j)\}$, we call $t_j$ $q$-enabled under marking $M$. This $q$ is denoted as $E(t_j | M)$.

Definition 6. $F = [f_1 f_2 \ldots f_m]^T$ is called an actual firing vector whose $j$-th element is $f_j = \min\{N(t_j | S(X)), E(t_j | M)\}$. $F$ can be partitioned into two parts as follows: $F = F_o + F_c$, where $F_c = [f_{c1} f_{c2} \ldots f_{cm}]^T$ is associated with effective conflict transitions, $F_o = [f_{o1} f_{o2} \ldots f_{om}]^T$ is associated with the other transitions. $F_o$ and $F_c$ satisfy the following conditions:

a. If $t_j$ is an effective conflict transition for $X$ under $M$, then $f_{oj} = 0$ and $f_{cj} = f_j$.

b. Otherwise, $f_{cj} = 0$ and $f_{oj} = f_j$.

3. Determination of the sufficient test space

If all the solutions of the equation $AX = M_d - M_0$ are tested, it can be found whether $M_d$ is reachable or not. But in some case, the solutions are infinite. Therefore, the tested range is determined in order to keep the method practical. This range must be finite and include at least one executable solution if it exists. This section will discuss how to determine the tested range.

Definition 7. Given the initial marking $M_0$ and the destination marking $M_d$ of a PN, $X$ is a solution of $AX = M_d - M_0$. If $M_d$ is reachable from $M_0$ under $X$, then $X$ is called an executable solution. Otherwise, $X$ is called a spurious solution.

Definition 8. $X = \{X_1, X_2, \ldots \}$ is the set of a solution $X$, the subset $X_e = \{X_{e1}, X_{e2}, \ldots \}$ of $X$ is called the sufficient test space if it satisfies following conditions:

i. If $M_d$ is reachable from $M_0$, there must exist at least one element in $X_e$ which is executable solution; in other words, if all elements in $X_e$ are not executable, then all the elements in $X$ are not executable either.

ii. $X_e$ is a finite set.

Definition 9. The vector $X$ which is a solution of $AX = 0$ is known as a T-invariant [18]. A solution $X$ is called positive if every element of $X$ is nonnegative.
**Definition 10.** The positive $T$-invariant solution $U$ of $AU=0$ is minimal if it satisfies the following condition: for any other $T$-invariant $U_i$, at least one element of $U-U_i$ is negative. The set of minimal $T$-invariant solutions is $U=\{U_1, U_2, \ldots, U_s\}$.

**Definition 11.** The positive particular solution $V$ of $AV=M_d-M_0$ is minimal if it satisfies the following condition: for any $T$-invariant $U$ of $PN$, there must be at least one element in $V-U$ which is negative, i.e. $\{U|V-U\geq 0, U$ is a $T$-invariant$\}=\emptyset$. The set of minimal particular solutions is $V=\{V_1, V_2, \ldots, V_q\}$.

The general solution of $AX=M_d-M_0$ must be expressed by the form of one minimal particular solution and the arbitrary linear combination of the $T$-invariant solutions as follows:

$$X = V_1 + \sum_{j=1}^{s} k_j U_j$$  \hspace{1cm} (2)

where $V_i \in V$, $k_j$ is nonnegative integer.

**Algorithm 1.** Interpretation of the computation for $X_e$.

**Step 1.** Solve the equation $AX=0$, get all the positive integer solutions $U=\{U_1, U_2, \ldots, U_s\}$, where each $U_j (1 \leq j \leq s)$ is a minimal $T$-invariant.

**Step 2.** Solve the equation $AX=M_d-M_0$, get all the positive integer particular solutions $V=\{V_1, V_2, \ldots, V_q\}$, where each $V_1 (1 \leq i \leq q)$ is a minimal particular solution. $B=\{B_1, B_2, \ldots, B_n\}$ is a subset of $V$.

If $V=\emptyset$, $M_d$ is not reachable, then end.

**Step 3.** Initialization: Let $X_e=V=\{V_1, V_2, \ldots, V_q\}$ and $X_{temp}=\Phi$.

If $U=\emptyset$, then end.

Otherwise, for every $V_i$, if $T(V_i)\cap T(U_j)$, then $V_i \not\subseteq B$. If $T(V_i)\not\subseteq T(U_j)$, then $V_i \subseteq B$.

Go to Step 4.

**Step 4.** For each pair of $(B_i, U_j)$, where $i=1, 2, \ldots |B|, j=1, 2, \ldots s$, and $|B|$ is the cardinality of set $B$, carry out the following operations:

If $\not\subseteq T(B_i) \cap T(U_j)$, choose the next pair of $(B_i, U_j)$.

If $\not\subseteq T(B_i) \cap T(U_j)$, if $\not\subseteq T(U_j) \subset T(B_i)$, choose the next pair of $(B_i, U_j)$.

If $\not\subseteq T(B_i) \cap T(U_j)$, if $\not\subseteq T(U_j) \not\subseteq T(B_i)$, then $D_i=\max(B_i)-U_j$, where $\max(B_i)$ is the maximum value of elements in $B_i$.

Let $D_i(r)$ be the $r$-th element of $D_i$.

$$W_i(r)=f(D_i(r)),$$ where $f(x)=\begin{cases} D_i(r), & \text{if } D_i(r) > 0 \\ 0, & \text{if } D_i(r) \leq 0 \end{cases}$, \hspace{1cm} r=1, 2, \ldots, m.

$$\sum_{r=1}^{m} (W_i(r) \cdot |p| p \in \not\subseteq T(U_j)) = \beta,$$ where $W_i(r)$ is the $r$-th element of $W_i$, $m=|T|$. 


Add \( B+k \cdot U \), \( k=1, 2, \ldots \beta \), to \( X_{\text{temp}} \).

When all pairs of \((B, U)\) have been tested, go to Step 5.

**Step 5.** If \( X_{\text{temp}}=\emptyset \), then end.

Otherwise, let \( B=X_{\text{temp}}, X_e=X_e \cup B \), \( X_{\text{temp}}=\emptyset \), go to Step 4.

Step 1 and Step 2 are to determine all the positive integer solutions \( X \) for equation \( AX=M_d-M_0 \). The firing count vector of any firing sequence from \( M_0 \) to \( M_d \) belongs to \( X \). In Step 4, if \( B \) is not an executable solution, then there must be some transitions in \( T(B) \) which aren’t enable it, i.e. some places in \( \circ T(B) \) are lack of tokens. In this case, if \( \{p\mid p \in \circ T(B) \cap T(U)\} \neq \emptyset \) and \( T(U) \not\subset T(B) \), then \( T(U) \) may provide tokens for \( \circ t \), where \( t \in T(B) \). Consequently, \( B+k \cdot U \) may be an executable solution, where \( k=1, 2, \ldots \beta \). Since the number of places and transitions in \( PN \) is finite, Step 4 and Step 5 only add finite elements to \( X_e \). Since the number of minimal T-invariants is finite, the finishing condition \( X_{\text{temp}}=\emptyset \), i.e. \( |\{p\mid p \in \circ T(B) \cap T(U)\} \neq \emptyset \) is satisfied after all the related T-invariants have been considered. As a result of the iterative process of Step 4→Step 5→Step 4, \( X_e \) includes at least one executable solution if it exists.

The following examples show how to implement the computation algorithm. These examples illustrate that suppressing any \( k \) in \( B+k \cdot U \), \( k=1, 2, \ldots \beta \), may eliminate some possible executable solutions.

**Example 1.** When the initial marking is \( M_0=(1,0,0,0,1,0,0,0,0) \) and the destination marking is \( M_d=(1,0,0,0,0,0,0,1) \) in Figure 1, calculate the sufficient test space \( X_e \). The \( \bullet \) and \( \circ \) symbols are represented as the initial and destination markings respectively.

**Step 1.** Solve the equation \( AX=0 \), get the positive integer minimal T-invariant \( U_1=(1,1,0,0,0,0,0,0) \).

**Step 2.** Solve the equation \( AX=M_d-M_0 \), get the positive integer minimal particular solution \( V=[V]=(0,0,0,0,1,1,1) \).

**Step 3.** Initialization: Let \( X_e=V, X_{\text{temp}}=\emptyset, B=X_e \).

**Step 4-1.** For \((V, U_1)\),

\[
\text{If } T(U_1) \subset T(V), \text{ then } D=V-\text{max}(V) \cdot U_1, \quad W(r)=W(D(r)),
\]

\[
\sum_{r=1}^{8} W(r) \|\{p\mid p \in \circ t \cap T(U_1)\} \| =3.
\]

Then add \( V+U_1, V+2 \cdot U_1, V+3 \cdot U_1 \) to the set of \( X_{\text{temp}} \).

Therefore, \( X_{\text{temp}}=\{V+U_1, V+2 \cdot U_1, V+3 \cdot U_1\} \).

**Step 5-1.** If \( X_{\text{temp}} \neq \emptyset \), then let \( B=X_{\text{temp}}=\{V+U_1, V+2 \cdot U_1, V+3 \cdot U_1\} \).

\( X_e=X_e \cup B=\{V, V+U_1, V+2 \cdot U_1, V+3 \cdot U_1\}, X_{\text{temp}}=\emptyset \).

Go to Step 4 in Algorithm 1.
Step 4-2. For any pair of \((B_i, U_j)\), \(T(U_j) \subseteq T(B_i)\) is satisfied. Therefore, \(X_{\text{temp}} = \emptyset\).

Step 5-2. If \(X_{\text{temp}} = \emptyset\), then end.

As a result of above sequence, \(M_d\) is reachable from \(M_0\). The firing sequence is \(t_5 \cdot t_1 \cdot t_2 \cdot t_6 \cdot t_7 \cdot t_3 \cdot t_4 \cdot t_8\). Its firing count vector corresponds to \(V + U_1 = (1, 1, 1, 1, 1, 1, 1, 1)\) in the sufficient test space \(X_e\). This example shows that suppressing \(B_{i+k} \cdot U_j\) \((k=1)\) in \(X_e\) may eliminate some possible executable solution.

Example 2. Consider the PN of Figure 2, given the initial marking \(M_0 = (1, 0, 0, 0, 0, 0, 0, 0, 1, 0)\) and the destination marking \(M_d = (0, 0, 0, 1, 0, 0, 0, 0, 1, 0)\), calculate the sufficient test space \(X_e\).
Step 1. Solve the equation $AX=0$, two positive integer minimal $T$-invariants are obtained:

$U_1=(0,0,0,1,1,1,0,0,1,1,0), \; U_2=(0,0,0,0,0,0,0,0,0,1,1)$

Step 2. Solve the equation $AX=M_a-M_0$, get the positive integer minimal particular solutions

$V=\{V\}={(1,1,1,0,0,0,0,1,0,0)}$

The general solution can be expressed as follows:

$X = (1,1,1,0,0,0,0,1,0,0) + k_1 \cdot (0,0,0,1,1,1,1,0,0,0) + k_2 \cdot (0,0,0,0,0,0,0,0,0,1,1)$

$k_1$ and $k_2$ are nonnegative integer.

Step 3. Initialization: Let $X_e=V$, $X_{\text{temp}}=\emptyset$, $B=X_e$

Step 4-1. For $(V, U_1)$, if $T(U_1) \not\subseteq T(V)$, then $D=V \cdot \max(V) \cdot U_1$, $W(r)=f(D(r))$,

$$\sum_{r=1}^{10} (W(r) \cdot |p| p \in T(U_1)) = 2.$$  

Then add $V+U_1, V+2U_1$ to $X_{\text{temp}}$. So $X_{\text{temp}}=\{V+U_1, V+2U_1\}$

For $(V, U_2)$, if $T(U_2) \not\subseteq T(V)$, then $D=V \cdot \max(V) \cdot U_2$, $W(r)=f(D(r))$,

$$\sum_{r=1}^{10} (W(r) \cdot |p| p \in T(U_2)) = 1.$$  

Then add $V+U_2$ to $X_{\text{temp}}$. So $X_{\text{temp}}=\{V+U_1, V+2U_1, V+U_2\}$

Step 5-1. If $X_{\text{temp}} \neq \emptyset$, then let $B=X_{\text{temp}}=\{V+U_1, V+2U_1, V+U_2\}$,

$X_e=X_e \cup B=\{V, V+U_1, V+2U_1, V+U_2\}$. Let’s put $X_{\text{temp}}=\emptyset$.

Go to Step 4 in Algorithm 1.

Step 4-2. For $(V+U_1, U_1)$, because $T(U_1) \subset T(V+U_1)$, choose the next pair.

For $(V+U_1, U_2)$, if $T(U_2) \not\subseteq T(V+U_1)$, then $D=(V+U_1) \cdot \max(V+U_1) \cdot U_2$, $W_1(r)=f(D_1(r))$,

$$\sum_{r=1}^{10} (W_1(r) \cdot |p| p \in T(U_2)) = 2.$$  

Then add $V+U_1+U_2$ and $V+U_1+2U_2$ to $X_{\text{temp}}$. So $X_{\text{temp}}=\{V+U_1+U_2, V+U_1+2U_2\}$

For $(V+2U_1, U_1)$, because $T(U_1) \subset T(V+2U_1)$, choose the next pair.

For $(V+2U_1, U_2)$, if $T(U_2) \not\subseteq T(V+2U_1)$, then $D=(V+2U_1) \cdot \max(V+2U_1) \cdot U_2$, $W_2(r)=f(D_2(r))$,
Then add $V+2U_1+U_2$, $V+2U_1+2U_2$, and $V+2U_1+3U_2$ to $X_{\text{temp}}$. So $X_{\text{temp}}=[V+U_1+U_2, V+U_1+2U_2, V+U_1+3U_2, V+U_2+3U_1]$

For $(V+U_2, U_2)$, because $T(U_2) \subset T(V+U_2)$, choose the next pair.

For $(V+U_2, U_1)$,

If $T(U_1) \subset T(V+U_2)$, then $D_3=(V+U_2)-\text{max}(V+U_2) \cdot U_1$, $W_3(r)=f(D_3(r))$,

$\sum_{r=1}^{10} \left| \{ p | p \in ^{o}t_i \cap T(U_2) \} \right| = 3.$

Then add $V+U_2+U_1$, $V+U_2+2U_1$, and $V+U_2+3U_1$ to $X_{\text{temp}}$. So $X_{\text{temp}}=[V+U_1+U_2, V+U_1+2U_2, V+U_1+3U_2, V+U_2+3U_1]$

**Step 5-2.** If $X_{\text{temp}} \neq \emptyset$, then let $B=X_{\text{temp}}=[V+U_1+U_2, V+U_1+2U_2, V+U_1+3U_2, V+U_2+3U_1]$.

So, $X_e=X_e+B=[V, V+U_1, V+2U_1, V+U_2, V+U_1+U_2, V+U_1+2U_2, V+2U_1+U_2, V+2U_1+2U_2, V+2U_1+3U_2, V+U_2+3U_1]$. Let’s put $X_{\text{temp}}=\emptyset$.

Go to Step 4 in Algorithm 1.

**Step 4-3.** For any pair of $(B_i, U_i)$, because $T(U_i) \subset T(B_i)$, $X_{\text{temp}}=\emptyset$

**Step 5-3.** If $X_{\text{temp}}=\emptyset$, then end

$M_\alpha$ is reachable from $M_\beta$. The firing sequence is $t_1^*t_2^*t_3^*t_1^*t_5^*t_6^*t_1^*t_2^*t_3^*t_5^*t_6^*t_10^*t_3^*t_3$. Its firing count vector corresponds to $V+2U_1+U_2=(1,1,1,2,2,2,1,1,1)$ in the sufficient test space $X_e$. This example illustrates that suppressing $B+i-kU_i$ ($k=\beta$) in $X_e$ may eliminate some possible executable solution.

### 4. Search of a firing sequence

Given the initial marking $M_\alpha$ and the destination marking $M_\beta$ of a PN, a solution $X_{\text{sol}}$ is solved from $AX=M_\beta-M_\alpha$. Then, an algorithm is developed to determine whether $M_\alpha$ is reachable from $M_\alpha$ under $X_{\text{sol}}$ or not. If $M_\alpha$ is reachable from $M_\alpha$, the algorithm gives the associated firing sequence $S(X_{\text{sol}})$.

**Definition 12.** Let $S=t_1t_2...t_r$ be a finite transition sequence. The transitions appearing in $S$ are defined by the set $Z(S) = \{t_1, t_2, ..., t_r\}$. The set of transitions $Z(S)$ is called a sequence component. $Z(S)$ is the set of elements that appear in a transition sequence $S$.

**Algorithm 2.** Search of a firing sequence $S(X_{\text{sol}})$ under $M_\alpha$

**Step 1.** According to $I$, determine all the conflict structure $C_i=<p, T_i>$, and construct $T_i$ and $C$. 
Reachability Criterion with Sufficient Test Space for Ordinary Petri Net

Step 2. Initialization: Let $M=M_0$, $X=X_{e_i}$, $S=\lambda$ ($\lambda$ is the sequence of length zero)


- If $F_o\neq0$, go to Step 4.
- If $F_o=0$ and $F_c\neq0$, go to Step 5.
- If $F=0$, go to step 6.

Step 4. If $F_o\neq0$, then there exists an $S(F_o)$ that has a firing sequence under $M$. Therefore, $S(F_o)$ can be fired. The reachable marking is calculated by $M'=M-A\cdot F_o$.

Let $M=M'$, $X=X-F_o$, $S=S\cdot S(F_o)$, where * is concatenation operation and $S\cdot S(F_o)$ means $S$ followed by $S(F_o)$. Go to Step 3.

Step 5. $F_o=0$ and $F_c\neq0$ means that all transitions in $S(F_c)$ are effective conflict transitions. Therefore, branching occurs and the number of branches is $|T(F_c)|$. From here, the computation has to consider all $|T(F_c)|$ branches.

After selecting a transition $t\in T(F_c)$, fire it, then the reachable marking is calculated by $M'=M-A\cdot e[t_j]$.

Let $M=M'$, $X=X-e[t_j]$, $S=S\cdot t_j$. Go to Step 3.

Step 6. If $X=0$, then $M_d$ is reachable from $M_0$ and $S=S(X_{e_i})$ is one of the firing sequences, end. Otherwise, go to Step 7.

Step 7. If all the branches in Step 5 have been implemented, then $M_d$ is not reachable, end. Otherwise, go to Step 5 and implement the remaining branches.

The validity of the above algorithm is proved as the following four cases:

Base: Let $X$ be a solution of $AX=M_d-M_0$. The actual firing vector $F=F_o+F_c$ is obtained with $M$ and $X$. Let $t\in T(F_o)$ and $t\in T(F_c)$.

Case 1: If $F_o\neq0$ and $F_c=0$, then multiple firing of $S(F_o)$ doesn’t affect a firing sequence associated with $X$ under $M_0$, for the input places of $T(F_o)$ don’t affect the enabling condition of other transitions in $T(X)$ except transitions in $T(F_o)$.

Case 2: If $F_o=0$ and $F_c\neq0$, then the firing of each transition in $S(F_c)$ is considered as a branch and implemented with respect to all branches. It means that all possibilities are involved. So, Algorithm 2 doesn’t eliminate any possible firing sequence.

Case 3: If $F_o=0$ and $F_c=0$, then no transition is enabled.

Case 4: If $F_o\neq0$ and $F_c\neq0$, then the multiple firing of $S(F_o)$ can be implemented before $S(F_c)$. It doesn’t eliminate any probability of finding a firing sequence associated with $X$ under $M_0$. It is proven in Proposition 1.

Proposition 1. If $\sigma\in S(X)$ is a firing sequence under $M_0$, then $(S(F_o)*\sigma')\in S(X)$ is a firing sequence under $M_0$ for any sequence $\sigma'$. 
Proof:

**Step 1.** Let \( T(F_0) = \{t_{o1}, t_{o2}, \ldots, t_{on}\} \). For a transition \( t_{o1} \in T(F_0) \), \( \sigma \) can be represented as \( \sigma = \sigma_1^{*} t_{o1}^{*} \sigma_2 \), where \( t_{o1} \notin Z(\sigma_1) \). Then \( M_0 \xrightarrow{\sigma_1} M_1 \xrightarrow{t_{o1}} M_2 \xrightarrow{\sigma_2} M_d \) is a firing sequence. Since \( T(F_0) \) is the set of transitions possible to be enabled under \( M_0 \), \( M_0 \) enables \( t_{o1} \). Therefore it is possible to put \( M_0 \xrightarrow{t_{o1}} M_3 \). By the definition of \( F_0 \), we have \( M_3(p) \geq M_0(p) \) for any \( p \in Z(\sigma_1) \). So \( \sigma_1 \) is enabled under \( M_3 \) because \( \sigma_1 \) is enabled under \( M_0 \) (Monotonicity Lemma). After \( \sigma_1 \) firing, \( M_2 \) is reachable from \( M_3 \). Therefore, we have \( M_0 \xrightarrow{\sigma_1} M_3 \xrightarrow{t_{o1}} M_2 \). Since \( \sigma_2 \) is enabled under \( M_2 \), \( t_{o1}^{*} \sigma_1^{*} \sigma_2 \) is a firing sequence under \( M_0 \).

**Step 2.** Under \( M_3 \), let’s consider the new \( T(F_0) = \{t_{o2}, \ldots, t_{on}\} \cup T(F_0') \), where \( T(F_0') \) is the set of transition generated after \( t_{o1} \) firing and may be empty. For a transition \( t_{o2} \in T(F_0) \), \( \sigma_1^{*} \sigma_2 \) can be represented as \( \sigma_1^{*} \sigma_2 = \sigma_3^{*} t_{o2}^{*} \sigma_4 \), where \( t_{o2} \notin Z(\sigma_3) \). Then \( \sigma_3^{*} t_{o2}^{*} \sigma_4 \) is a firing sequence. By the same way described in Step 1, we can prove that \( t_{o2}^{*} \sigma_3^{*} \sigma_4 \) is a firing sequence under \( M_3 \).

**Step 3.** By Step 1 and Step 2, \( t_{o1}^{*} t_{o2}^{*} \sigma_3^{*} \sigma_4 \) is a firing sequence under \( M_0 \).

**Step 4.** In the same way, it is proven that \( t_{o1}^{*} t_{o2}^{*} \cdots t_{on}^{*} \sigma_i^{*} \sigma_j \) is a firing sequence under \( M_0 \).

According to the definition of \( F_0 \), all transitions in \( \{t_{o1}, t_{o2}, \ldots, t_{on}\} \) can fire simultaneously under \( M_0 \). Let’s put \( \sigma' = \sigma_i^{*} \sigma_j \), then \( (S(F_0) \sigma') \in S(X) \) is a firing sequence under \( M_0 \).

**Example 3.** Let us now apply the proposed algorithm to the PN of Figure 3. Given \( M_0 = (0,0,0,0,0,1,0,0) \), \( M_d = (0,0,0,0,1,0,1,0) \) and \( X = (1,2,1,1,1,1,1) \), determine if \( M_d \) is reachable or not under \( M_0 \) and \( X \).

![Figure 3. Petri net structure](image-url)
1. There are two conflict structures, \( C_1 = (p_1, \{t_2, t_5\}) \), \( C_2 = (p_2, \{t_3, t_4\}) \), \( T_c = \{t_2, t_3, t_4, t_5\} \).

2. Initialization: \( M = M_0 = (0,0,0,0,0,0,0,0,0) \), \( X = (1,2,1,1,1,1) \), \( S = \lambda \).

3. Under \( M \) and \( X \), only \( t_6 \) is 1-enabled. Then, \( F_0 = (0,0,0,0,0,1) \).

4. Fire \( S(F_0) = t_6 \). Then the reachable marking \( M' \) becomes \((1,0,0,0,0,0,0,0,0)\).

Let \( M = M' \), \( X = X - F_0 = (1,2,1,1,1,0,0) \), \( S = t_6 \). Go to Step 3 in Algorithm 2.

3-1. Under \( M \) and \( X \), \( F_0 = (0,0,0,0,0,0,1) \).

4-1. Fire \( S(F_0) = t_7 \). Then, the reachable marking becomes \( M' = (1,0,0,0,0,1,0,0,0) \).

Let \( M = M' \), \( X = X - F_0 = (1,2,1,1,1,0,0) \), \( S = t_6 \). Go to Step 3 in Algorithm 2.

3-2. Under \( M \) and \( X \), \( F_0 = (0,1,0,0,0,0,0) \). Go to Step 4 in Algorithm 2.

4-2. Fire \( S(F_0) = t_2 \) (\( t_2 \) is not an effective conflict transition because \( t_5 \) cannot enable),

then the reachable marking becomes \( M' = (0,1,0,0,1,0,1,0,1) \).

Let \( M = M' \), \( X = X - F_0 = (1,1,1,1,1,0,0) \), \( S = t_6 \). Go to Step 3 in Algorithm 2.

3-3. Under \( M \) and \( X \), \( F_0 = (0,0,1,0,0,0,0) \). Go to Step 4 in Algorithm 2.

4-3. Fire \( S(F_0) = t_3 \) (\( t_3 \) is not an effective conflict transition because \( t_4 \) cannot enable),

then the reachable marking becomes \( M' = (0,0,1,0,1,0,1,0,1) \).

Let \( M = M' \), \( X = X - F_0 = (1,1,0,1,1,0,0) \), \( S = t_6 \). Go to Step 3 in Algorithm 2.

3-4. Under \( M \) and \( X \), \( F_0 = (0,0,0,0,0,0) \). Go to Step 6 in Algorithm 2.

6. Because \( X \neq 0 \), go to Step 7 in Algorithm 2.

7. There is no effective conflict transition i.e., no branch. Consequently, \( M_d \) is not reachable under \( X \) because \( X \neq 0 \).

The above implementing process can be presented by a firing path tree as shown in Figure 4.

5. Application of Reachability Criterion

An example will be given to illustrate how to use the proposed method of Algorithm 1 and Algorithm 2 to solve the reachability problem.

Example 4. When the initial marking is \( M_0 = (1,0,0,0,0,0,0,0,0,1) \) in the PN of Figure 5, is the destination marking \( M_d = (0,0,1,0,1,0,0,0,1) \) reachable from \( M_0 \)?

First, calculate sufficient test space using the following steps:

1. Solve the equation \( AX = 0 \), get one positive integer minimal \( T \)-invariant

\[
U = (0,0,0,0,0,0,0,0,1,1).
\]

2. Solve the equation \( AX = M_0 - M_0 \), get the positive integer minimal particular solutions

\[
V_1 = (0,2,1,0,2,0,0,0), \quad V_2 = (2,2,1,2,0,0,0,0) \text{ and } V_3 = (1,2,1,1,1,0,0,0)
\]

3. Initialization: Let \( X_e = \{V_1, V_2, V_3\} \), \( X_{temp} = \emptyset \), \( B = X_e \).

4-1. For \( (V_i, U) \),
Figure 4. Firing path tree on reachability of Figure 3.
Figure 5. Petri net structure

If \( T(U) \subset T(V_1) \), then \( D_1 = V_1 - \text{max}(V_1) \cdot U, W_1(r) = f(D_1(r)) \),
\[
\sum_{r=1}^{8} |\{ p \mid p \in t_i \cap T(U) \}| = 2
\]

Then add \( V_1 + U, V_1 + 2U \) to \( X_{\text{temp}} \). Then, \( X_{\text{temp}} = \{ V_1 + U, V_1 + 2U \} \)
For \( (V_2, U) \), because \( ^{o}T(V_2) \cap T(U) = \Phi \), choose the next pair.
For \( (V_3, U) \),
If \( T(U) \subset T(V_3) \), then \( D_3 = V_3 - \text{max}(V_3) \cdot U, W_3(r) = f(D_3(r)) \),
\[
\sum_{r=1}^{8} |\{ p \mid p \in t_i \cap T(U) \}| = 1
\]

Then add \( V_3 + U \) to \( X_{\text{temp}} \), \( X_{\text{temp}} = \{ V_1 + U, V_1 + 2U, V_3 + U \} \)

Step 5-1. If \( X_{\text{temp}} \neq \Phi \), then let \( B = X_{\text{temp}} = \{ V_1 + U, V_1 + 2U, V_3 + U \} \),
\( X_e = X_e \cup B = \{ V_1, V_2, V_3, V_1 + U, V_1 + 2U, V_3 + U \} \). Let’s put \( X_{\text{temp}} = \Phi \). Go to Step 4 in Algorithm 1.

Step 4. For any pair of \( (B_i, U) \), because \( T(U) \subset T(B_i) \), \( X_{\text{temp}} = \Phi \).

Step 5. If \( X_{\text{temp}} = \Phi \), then end.

Consequently, the sufficient test space becomes \( X_e = \{ V_1, V_2, V_3, V_1 + U, V_1 + 2U, V_3 + U \} \).
Second, calculate a firing sequence in order to test if \( M(d) \) is reachable from \( M(0) \) under some element in \( X_e \).
The elements of the sufficient test space \( X_e \) are calculated separately as follows:

Step 1. For \( X = V_1 = (0,2,1,0,2,2,0,0) \)

The implementing process is shown in Figure 6.
Step 2. For $X = V_2 = (2,2,1,2,0,0,0,0)$

Carrying out the same process, the conclusion is as follows: $M_d$ is not reachable under $V_2$.

Step 3. For $X = V_3 = (1,2,1,1,1,0,0,0)$

Carrying out the same process, the conclusion is as follows: $M_d$ is not reachable under $V_2$.

Step 4. For $X = V_1 + U = (0,2,1,0,2,2,1,1)$

Carrying out the same process shown in Figure 7, the conclusion is as follows: $M_d$ is reachable from $M_0$ under $V_1 + U$. $V_1 + U$ is an executable solution in $X_e$ and the firing sequence is $t_5 * t_7 * t_6 * t_2 * t_3 * t_5 * t_6 * t_2 * t_8$.

As a result of calculating each element of the sufficient test space $X_e = \{V_1, V_2, V_3, V_1 + U, V_1 + 2U, V_3 + U\}$ individually, a firing sequence is finally found at the fourth element ($V_1 + U$) of $X_e$. Therefore, the elements $V_1 + 2U$ and $V_3 + U$ don’t need to be calculated. Consequently, the structure of the Petri net (Figure 5) is shown to possess at least one reachable firing sequence.
6. Conclusions

In this chapter, a new general criterion has been created to solve the reachability problems for ordinary Petri nets. This criterion is based on two processes: (i) Calculating the sufficient test space. (ii) Testing whether or not the destination marking is reachable from the initial marking under the sufficient test space. The sufficient test space significantly reduces the quantity of computation needed to search for an executable solution in X. The firing path tree shows the firing sequence of an executable solution. Consequently, if the destination marking is reachable from the initial marking, this method gives at least one firing sequence that leads from the initial marking to the destination marking. Some examples are given to illustrate how to use this method to solve the reachability problem. This algorithm can be utilized in the following fields: Path searching, auto routing, and reachability between any places in a complicated network.

Author details

Gi Bum Lee

*Research Institute of Industrial Science & Technology, Pohang, Korea*
7. References


